# УЕБ БАЗИРАНА ПАРАЛЕЛИЗИРАНА СИСТЕМА ЗА РЕШАВАНЕ НА АЛГЕБРИЧНИ УРАВНЕНИЯ ОТ ПРОИЗВОЛНА СТЕПЕН[1]

***Антон Илиев*'**, Николай Кюркчиев**, Никола Вълчанов*'**, Тодорка Терзиева*, Тодор Тодоров****

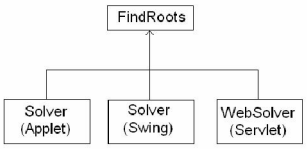
**4003 Пловдив, бул. „България” 236,*
*Пловдивски университет „Паисий Хилендарски”,*
*Факултет по математика и информатика*
***1113 София, ул. Акад. Георги Бончев, бл. 8,*
*Институт по математика и информатика,*
*Българска академия на науките*
*e-mail: aii@uni-plovdiv.bg, nkyurk@math.bas.bg, nvalchanov@gmail.com,*
*dora@uni-plovdiv.bg, tod_bul@hotmail.com*

*Решаването на алгебрично уравнение се свежда до използване на числени методи, чрез които се намират корените с известно приближение. Един дял от методите за решаване на такива уравнения е паралелни (или съвместни, или наричани още едновременни от английското simultaneous) методи [1] за определяне на всички корени на алгебрично уравнение. В настоящата статия е описано приложение, имплементиращо такъв метод. В литературата липсва информация за друг солвър използващ паралелни методи [1] за решаване на уравнения.*

***Ключови думи:*** *компютърно решаване на уравнения, алгоритми.*

I. Структура на системата. Описание на модулите.

Системата е реализирана на JAVA 5 като три многонишкови приложения: 1) Мрежово – server side, като servlet; 2) Мрежово – client side, като applet; и 3) Самостоятелно swing приложение. Системата е написана на Java поради платформената ѝ независимост. Всеки един от трите варианта има своите предимства и недостатъци. Различните части на системата могат да се представят със следната диаграма:




FindRoots.java е ядрото на системата и съдържа всичко необходимо за решаване на уравнения. Сам по себе си предоставя съвсем прост конзолен интерфейс, но е основата на по-сложни приложения. В тази част се извършват най-тежките изчисления поради което изчислението е реализирано многонишково като всяка нишка изчислява един корен. По този начин се постига мащабируемост и на компютър (най-често сървър) с повече от един процесор се постига по-добра производителност. В последните години цените на многоядрените процесори паднаха значително и вече се продават само такива.

Solver.java имплементира графичен интерфейс към системата и дава възможност интерактивно да се наблюдава решаването на всяка стъпка на уравнението, както графично така и в текстова част се показват текущите стойности. Solver може да работи като локално swing приложение или като client side Java Applet.

WebSolver.java е сървърно разширение реализирано като Servlet. Резултатът тук се показва само текстово и едва след като решението е намерено, но за сметка на това се изпълнява на сървъра и не заема ресурси на клиента.

II. Описание на класовете.

| Клас/интерфейс | Описание |
|---|---|
| **SolverEvent** | Интерфейс, наследниците на който реализират callback методи |
| **SolverNullEvent** | Конкретен наследник на **SolverEvent,** реализиращ поведение по подразбиране – т.е. нищо. |
| **MathStuff** | Общи математически операции |
| **Complex** | Операции с комплексни числа |
| **MyRect** | Описва правоъгълна област |
| **Root** | Представя корен с неговата кратност |

[1] Тази работа е финансирана по проект ИСМ-4 към поделение „Научна и приложна дейност” на ПУ „Паисий Хилендарски”.

| | |
|---|---|
| **Eq** | Помощен клас описващ списък от подобласти |
| **TPt** | Представя комплексно число като точка, с ъгъл и квадрант на нейната функционална стойност |
| **FindRoots** | Решава уравнение с локализация на Cauchy [1] |
| **LocalizeSec** | Решава уравнение ползвайки едновременен метод [1] |

| **Interface SolverEvent** | |
|---|---|
| public void update(Complex[] roots); | Callback метод извикван след всяко преизчисляване на корените |

| **class SolverNullEvent implements SolverEvent** | |
|---|---|
| public void update(Complex[] roots); | Callback метод извикван след всяко преизчисляване на корените |

| **class MathStuff** | |
|---|---|
| public static int arcTan(BigDecimal tan) | Намира аркус тангенс с точност до 10 градуса. |

| **class Complex** | |
|---|---|
| public BigDecimal getRe() | Извлича реалната част |
| public BigDecimal getImg() | Извлича имагинерната част |
| public String toString() | Преобразува комплексното число до низ |
| public String toString2() | Преобразува комплексното число до низ, ако е достатъчно близо до 0 го закръглява до 0 |
| public String toString3() | Преобразува комплексното число до низ с понижена точност, ако е достатъчно близо до 0 го закръглява до 0 |
| public Complex divide(Complex dvs, MathContext mc) | Операция делене на комплексно число с дадена точност |
| public Complex divideReal(BigDecimal dvs, MathContext mc) | Делене на реално число с определена точност |
| public Complex add(Complex a) | Добавяне на комплексно число |
| public Complex addReal(BigDecimal a) | Добавяне на реално число |
| public Complex sub(Complex a) | Изваждане на комплексно число |
| public Complex mul(Complex a) | Умножаване на комплексно число |
| public BigDecimal absCSqr() | Модул на комплексното число повдигнат на квадрат |
| public BigDecimal distSqr(Complex a) | Разстояние между две числа повдигнато на квадрат |
| public Complex round(MathContext mc) | Закръгляване на комплексно число с определена точност |
| public int kvadrant() | Връща квадрант на аргумента |
| public int kvadrantEx(BigDecimal[] ft) | Разширен квадрант на функционалната стойност |
| public int getAngle(FindRoots fr) | Ъгъл на функционалната стойност |
| Complex mid(Complex x) | Средна точка |
| boolean checkBigStep(Complex y) | Проверява дали разликата между две числа е поне два пъти |
| public boolean inRgn(MyRect rct) | Проверява дали числото е в дадена област |

| | |
|---|---|
| public boolean isInLimit (Complex bound, Limit limit) | Дали числото е напуснало определена граница |

| **class MyRect** | |
|---|---|
| public MyRect copy() | Прави копие на областта. |
| public String toString() | Връща низ описващ областта |

| **class Eq** | |
|---|---|
| public void compact() | Премахва областите без корен. |
| public int getRgns(FindRoots fr) | Прави хоризонтално или вертикално разделяне |
| public int getRgnsHoriz(FindRoots fr) | Прави хоризонтално разделяне |
| public int getRgnsVert(FindRoots fr) | Прави вертикално разделяне |

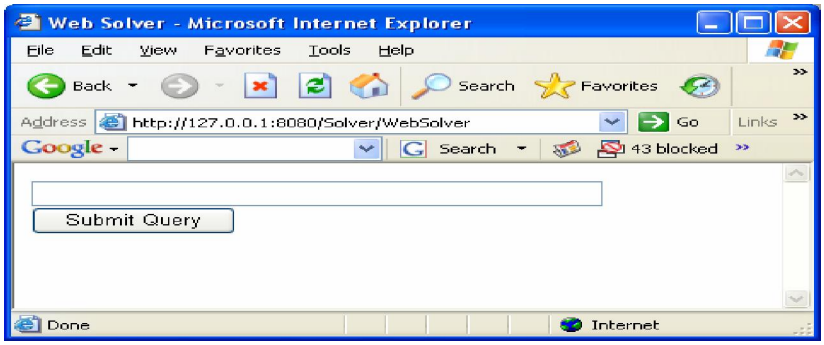
| **class FindRoots** | |
|---|---|
| public BigDecimal getRoot(BigDecimal xx, int n) | Извлича корен n от xx |
| public Complex fx2(Complex xx) | Намира f(x) |
| public Complex fproizv(Complex fv, int proizv) | Намира производната proizv на f(x) |
| public BigDecimal getRadius() | Намира радиуса на уравнението според Fujiwara [1,2] |
| private void buildPoints (ArrayList<TPt> pts, Complex start, Complex end, Limit limit) | Намира всички точки за локализация за дадена права |
| public int rectanEx(MyRect rgn, boolean firstIter) | Определя броя корени в дадена област |
| private Complex sum(Eq eq, int i13) | Помощен метод на prove |
| private boolean checkRoot(Complex r) | Проверява дали r е корен |
| private boolean prove(Eq eq) | Намира корен |
| public void fillKoef(String eqtn) | Инициализира уравнението с низа eqtn |
| public String getEquation() | Връща уравнението като низ |
| public void fRoots(String eqtn, SolverEvent evt) | Смята уравнението |

| **class LocalizeSec** | |
|---|---|
| public void init() | Инициализира локализацията |
| public double localizeStep (MathContext mc) | Извършва стъпка от локализацията |
| private boolean proveSec() | Намира корени |
| public void solve (String eqtn, SolverEvent evt) | Смята уравнението |

**III. Потребителски интерфейс.**

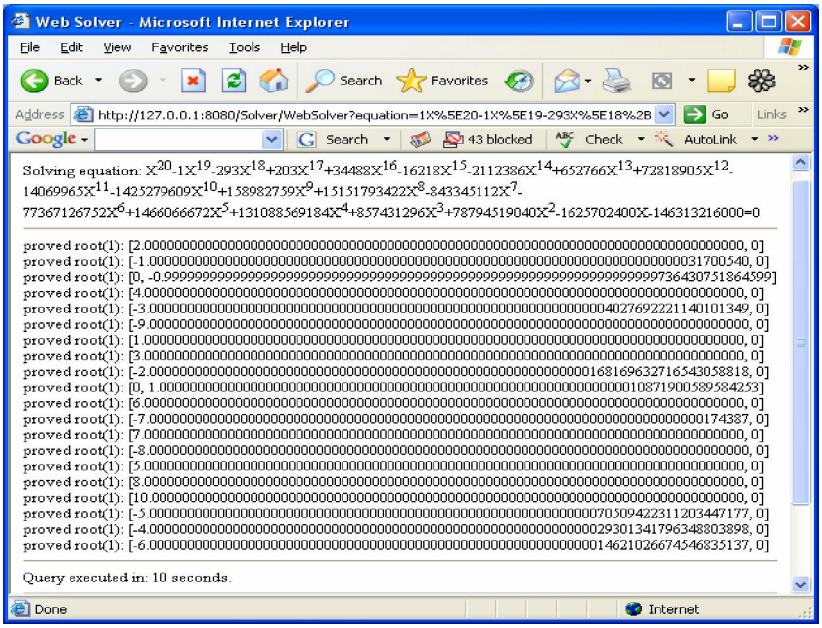
В системата потребителския интерфейс е сравнително опростен. На входа потребителя въвежда уравнение от вида: $a_0x^n + a_1x^{n-1} + \ldots + a_{n-1}x + a_n$ (Уравнението се въвежда без „=0”) без да е необходимо въвеждане на начални приближения. Applet и Swing версиите са напълно еквивалентни, поради което ще разглеждаме само Applet.

**Servlet** [3-5]. Ако инсталацията е коректна с web browser се стартира сървлета чрез URL от вида: **http://localhost/solver/WebSolver**

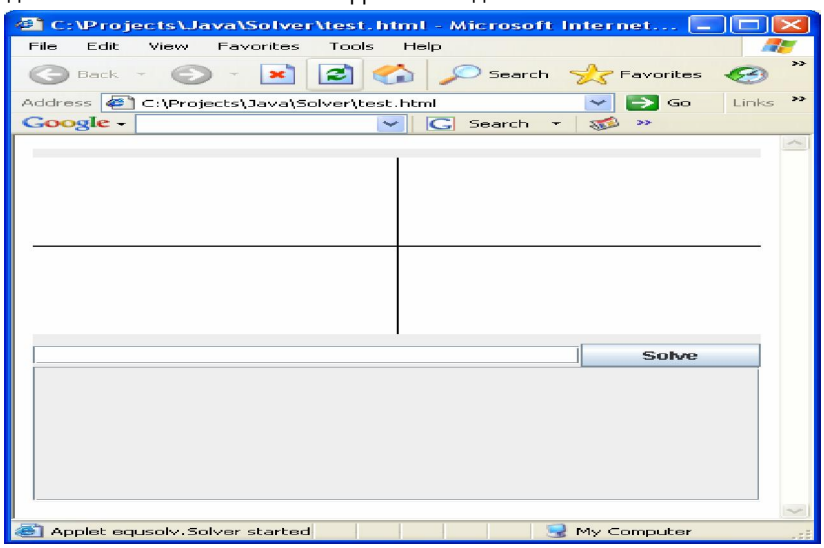



Въвеждаме уравнението и след натискане на бутона „Submit Query” се показват последователно самото уравнение, корените с техните кратности и времето за изпълнение на заявката. След това отново има поле за въвеждане на ново уравнение.

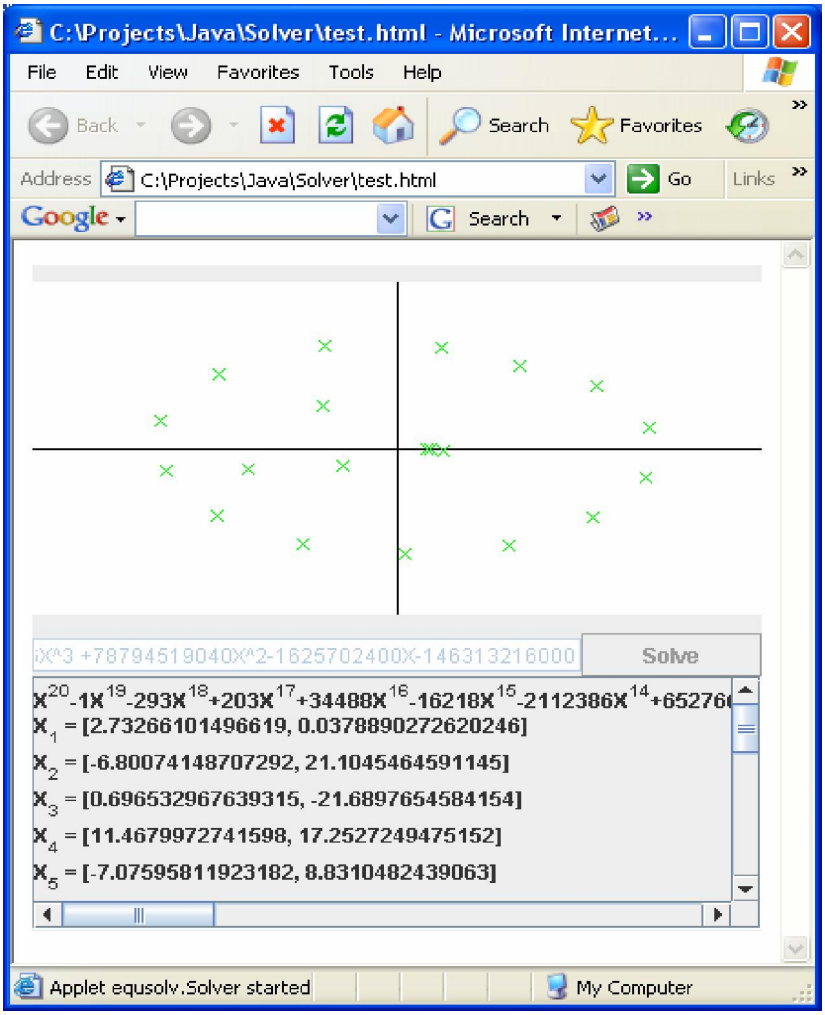
За пълнота ще приложим и решението на едно не толкова тривиално уравнение



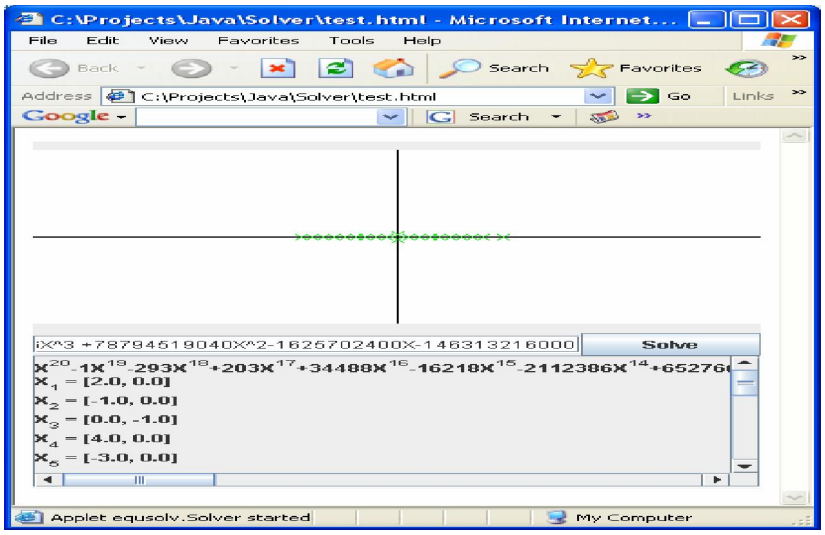

**Applet** и **Swing** [3-5]. Тези два варианта не се отличават много по функционалност, като изключим по-доброто визуално представяне. В начално състояние applet изглежда така:




След въвеждане на уравнението и натискане на бутон „Solve”, двете контроли стават сиви докато не приключи изчислението, в процеса на пресмятане на екрана се обновяват текущите стойности:




По време на текущото изчисление процесът на търсене се изобразява не само графично в горната част, но също така и текстово. В случаи като показания, когато мястото е недостатъчно се появяват ленти за скролиране, които са достъпни по всяко време. След приключване на изчислението решението изглежда по следния начин:




След завършването солвъра дава 18 реални и 2 имагинерни корена, които сме представили във вида [Re,Im], където с Re е означена реалната част на корена, а с Im съответната имагинерна част: [1,0], [-2,0], [6,0], [7,0], [3,0], [10,0], [8,0], [-4,0], [5,0], [-9,0], [-3,0], [-8,0], [2,0], [0,1], [-6,0], [-7,0], [-5,0], [0,-1], [4,0] и [-1,0].

Намирането на корените на уравнение е стара, но и до днес не съвсем лесно решима задача. Системата е тествана както под Windows XP, така и под Linux (Fedora). Въпреки, че системата върши добре своята работа на машина с повече от един процесор като възможност за развитие може да се

посочи разпределена работа на няколко машини чрез RMI. Допълнително развитие може да има във връзка с оптимизация на системата, както и по-ефективна синхронизация на нишките.

**Литература**

[1] Iliev, A., N. Kyurkchiev, T. Todorov. Web-Based Simultaneous Equation Solver, International Journal Information Theories & Applications, 2003, 10, № 4, 468-471.

[2] Obreshkoff, N. On the numerical solving equations. Ann. of the Sofia University, 1963, 56, 73-83.

[3] http://java.sun.com/javaee/5/docs/tutorial/doc - Онлайн книга за Java™ EE 5.

[4] Heller, Ph., S. Roberts. Complete Java 2® Certification: Study Guide, fifth edition, 2005.

[5] Ball, J., D. Carson, I. Evans, S. Fordin, K. Haase, E. Jendrock. The Java™ EE 5 Tutorial – Sun Microsystems, 2006.